



METHOD AND APPARATUS FOR MEASURING ONE-WAY
DELAY AT ARBITRARY POINTS IN NETWORK

10

This invention is a continuation in part of U.S. Patent Application Ser. No.
10/723,118, filed November 26, 2003, entitled: Method and Apparatus For Analyzing a
Media Path in a Packet Switched Network.

15

BACKGROUND

A User Datagram Protocol (UDP) traceroute packet is currently used to identify
problems in Internet Protocol (IP) networks. The traceroute packet is sent into the IP network
with a small TTL value. The small TTL value is decremented by each intermediate router
that forwards the traceroute packet through the IP network. The intermediate router that
20 decrements the TTL value to zero refuses to forward the traceroute packet and replies back to
the traceroute packet source with an Internet Control Message Protocol (ICMP) 'TTL
exceeded' message.

Traceroute packets are not effective in measuring network delay. For example, the
'TTL exceeded' message does not indicate when the intermediate node received the traceroute
25 packet. Further, the 'TTL exceeded' message may travel back to the source through a
different path than the path used for originally sending the traceroute packet. Therefore, the
round trip delay from when the traceroute packet was originally sent to when the 'TTL
exceeded' message is received back at the source does not identify the one-way packet delay
for the traceroute packet.

30

The present invention addresses this and other problems associated with the prior art.

SUMMARY OF THE INVENTION

Timestamps are inserted into trace packet expiration messages to identify delay in a network. A Time To Live (TTL) value in the trace packet is varied to intentionally cause an intermediate node in the network to discard the trace packet and send back the packet expiration message. The intermediate node sending the packet expiration message inserts a
10 time value in the message indicating when the intermediate node received the trace packet. The time value is then used to determine the one-way delay time required for the trace packet to reach the intermediate node.

The foregoing and other objects, features and advantages of the invention will become more readily apparent from the following detailed description of a preferred embodiment of
15 the invention which proceeds with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1A is a diagram of a system that identifies one-way delay at arbitrarily selected nodes in an Internet Protocol (IP) network.

20 FIG. 1B is a diagram showing why the system in FIG. 1A sends back fault notices that include timestamps.

FIG. 1C is a flow diagram showing how an intermediate node processes a received trace packet.

FIG. 1D is a diagram of a system in another embodiment that uses media trace
25 packets to identify problems in a media path.

FIG. 2 is a block diagram showing in more detail how the media trace packets in FIG. 1D operate in a media path.

5 FIG. 3A is a flow diagram showing how the media trace packets are processed at an endpoint.

FIG. 3B is a flow diagram showing another embodiment of how the media trace packets are processed at an endpoint.

FIG. 4 is a diagram of a Real Time Protocol (RTP) media trace packet.

10 FIG. 5 is a block diagram showing how no-op payload packets are used in combination with media trace packets.

FIG. 6 is a block diagram showing how the no-op packet evaluation scheme is negotiated between two endpoints.

FIG. 7 is a flow diagram showing how the no-op payload packets are generated.

15 FIG. 8 is a flow diagram showing how a media path is analyzed using the no-op media packets.

FIG. 9 is a flow diagram showing how analysis information about the media path is used for establishing the media path.

FIG. 10 is a diagram of an RTP no-op packet.

20 FIG. 11 is a detailed diagram of a network device that processes the no-op packets and the media trace packets.

DETAILED DESCRIPTION

Calculating One-Way Delay

25 Referring to FIG. 1A, trace packets 108 and 110 are used for analyzing one-way packet delay in a Wide Area Network 94. A source endpoint 96 may send out trace packets 108 and 110 directed to a same destination endpoint 98 or to different destination endpoints. The source endpoint 96 intentionally sets specific Time To Live (TTL) values in the trace

5 packets that cause intermediary nodes in the WAN 94 to discard the trace packets and send back a time to live exceeded message prior to the trace packets reaching the destination endpoint 98.

The time to live exceeded messages are alternatively referred to below as fault notices 104 and 106. Timestamp values T1 and T2 are inserted into the fault notices 104 and 106,
10 respectively, indicating when the trace packets 108 or 110 were received by the intermediate nodes 100 and 102. The fault notices 104 and 106 are then sent by the intermediate nodes 100 and 102, respectively, back to the source endpoint 96. The source endpoint 96 then uses the timestamp values in the fault notices 104 and 106 to calculate one-way packet delays between the source endpoint 96 and the intermediate nodes 100 and 102. .

15 In one example, source endpoint 96 sends out the first trace packet 108 with a TTL value of one. The first intermediate node 100 receiving the trace packet 108 decrements the TTL value by one to zero. This causes the intermediate node 100 to decide to send a fault notice 104 back to the source endpoint 96 indicating the TTL value has been decremented to zero. The intermediate node 100 records the time the packet was received, and inserts this
20 time value T1 into the fault notice 104. The time T1 indicates when the trace packet 108 was received by intermediate node 100.

The current time T1 can be derived from an internal reference time-of-day clock maintained by intermediate node 100. The current time associated with intermediate node 100 can also be periodically updated by a network synchronized time value received over the
25 network 94. For example, the timestamp value T1 can be maintained using the Network Time Protocol (NTP).

The source endpoint 96 may then send out the second trace packet 110 with a TTL value of, for example, two. The first intermediate node 100 receiving the trace packet 110

5 decrements the TTL value by one to one. Because the TTL value is not zero, the intermediate node 100 does not send back a fault notice in response to trace packet 110. Instead, the intermediate node 100 forwards the trace packet 110 on to a next hop in WAN 94. In this example, intermediate node 100 forwards the trace packet 110 to intermediate node 102.

10 The second intermediate node 102 decrements the TTL value in trace packet 110 by one to zero. This causes the intermediate node 102 to decide to send the fault notice 106 back to the source endpoint 96 indicating the TTL value in trace packet 110 has been decremented to zero. The intermediate node 102 records the time the packet was received, and inserts this time value T2 into fault notice 106 that indicates when the trace packet 110
15 was received. The current time T2 represents the one-way delay time required for trace packet 110 to go from source endpoint 96 to intermediate node 102. Intermediate node 102 would also maintain a time-of-day clock, as described above for intermediate node 100.

The source endpoint 96 keeps track of the send times when the trace packets 108 and 110 were both sent from source endpoint 96. The source endpoint 96 then compares the time
20 values T1 and T2 received from intermediate nodes 100 and 102, respectively, with the send times to determine one-way delay times for trace packets 108 and 110. For example, the one-way delay from source endpoint 96 to intermediate node 102, is the time T2 minus the send time when the source endpoint 96 initially sent out trace packet 110. Much like nodes 100 and 102, source endpoint 96 needs to also maintain an accurate time of day clock.

25 FIG. 1B shows one reason why the timestamp values T1 and T2 are inserted into the fault notices 104 and 106, respectively. The fault notices 104 and 106 may be sent back to the source endpoint 96 over paths that are different both from the path used for routing the trace packets and different from the path taken by fault notices sent by other intermediate

5 nodes. For example, the fault notice 104 sent by intermediate node 100 may be routed back to source endpoint 96 over a path that includes network nodes 112A and 112B. The fault notice 106 sent by intermediate node 102 may be routed back over a different path that includes network nodes 114A-114C.

Different times may be required for the two fault notices 104 and 106 to arrive back to the source endpoint 96. Simply measuring the time required for the fault notices 104 and 106 to arrive back to the source endpoint 96 is not a reliable way to identify network delay. The fault notices 104 and 106 identify the time that the intermediate nodes received the trace packets. Thus, the one-way delay for the trace packets can be determined using the fault notice time values independently of when the fault notices 104 and 106 arrive back at the source endpoint 96.

The endpoints 96 and 98 can include any combination of hardware and software for receiving or sending a trace packet over the WAN 94. For example, the endpoints 96 and 98 can comprise computers, servers, gateways, switches, Voice over Internet Protocol (VoIP) phones, or any other network processing device. The intermediate nodes 100 and 102 are any network processing device, such as routers, switches, gateways, etc. capable of forwarding packets and sending back a fault notice.

FIG. 1C is a flow diagram that shows in further detail how intermediate nodes process the trace packets. In block 120, an intermediate node receives a trace packet. The intermediate node decrements the TTL value in the trace packet in block 122. If the TTL value is not decremented to zero in block 124, the trace packet is forwarded to a next network hop toward a destination endpoint in block 132.

If the TTL value for the trace packet is decremented to zero in block 124, the intermediate node acquires a current time in block 126. The current time is inserted into a

fault message in block 128 and the fault message sent back to the trace packet sender in block 130. The intermediate node may append certain information from the trace packet into the fault message in block 128. For example, the node could append information from the trace packet header or information from the trace packet payload.

In one example, the network device sending the trace packet could include a sending time value in the trace packet. The intermediate node sending the fault message could include the sending time from the trace packet with the local time value when the trace packets was received in the fault message. Then the fault message would have all the information required for calculating the one-way trace packet delay. In another example, the network device sending the trace packet could simply place monotonically-increasing values into the trace packets and locally store the time each packet was sent.

In another embodiment, the intermediate node or the sending device 96 could send the calculated one-way delay time to a server or computer in addition to sending the ICMP messages back to the source endpoint originally sending the trace packet. That would enable fault notices from different source endpoints to be monitored by a same central network administration device.

In one specific embodiment, the intermediate nodes insert a 32-bit Network Time Protocol (NTP) time in Internet Control Message Protocol (ICMP) messages. The ICMP messages are discussed in Request for Comment (RFC) 792 which is herein incorporated by reference. The source endpoint also includes a NTP-synchronized time clock which can be used to compare trace packet sending times with the timestamp values returned in the trace packet TTL time expired notices.

Several ICMP (RFC792) messages have a 32-bit field that is defined as "unused". The intermediate node inserts the NTP time into this currently unused field. The ICMP

messages with unused fields that are used for carrying the NTP time can include Destination Unreachable Messages, Time Exceeded Messages, and Source Quench Messages.

The ICMP messages is used by the intermediate nodes to indicate the TTL time has expired and the destination endpoint is unreachable. The NTP time is inserted into the ICMP message to allow calculation of one-way delay. Thus, a system is provided that is capable of measuring one-way delay to arbitrary network devices in a network.

Media Trace Packets

5 In one example, the trace packets are used in conjunction with media trace packets as described below. This is just one example of how the timestamps can be used in a media path and it should be understood that the trace packet timestamps can be used in any network application.

Referring to FIG. 1D, media trace packets 203 are used for analyzing a media path
10 212 in an Internet Protocol (IP) network 202. The media trace packets 203 have the same information in header 204 as regular media payload packets. Therefore, the media trace packets 203 will travel along the same media path 212 as media packets that are actually carrying a media payload. Specific Time To Live (TTL) values are set in the header TTL field 206 that intentionally cause faults in intermediary nodes, such as node 213, in the media
15 path 212. The intermediary node 213 in response sends a fault notice 215 back to the source endpoint 200. The fault notice 215 is used for conducting QoS analysis of the media path 212.

In one embodiment, the intermediate node 213 includes the local timestamp value in fault notice 215 as described above in FIGS. 1A-1C so the source endpoint 200 can calculate
20 one-way delay times in media path 212.

5 One application for the media trace packet 203 is in the Real Time Transport Protocol (RTP), which is described in RFC 3550. The RTP protocol is used for carrying a real time media stream over the IP network 202. The media stream can contain any type of real time media, including audio, video, real time text transmission, or voice-band data such as fax or modem information.

10 In one example, the source endpoint 200 initiates a RTP media call to a destination endpoint 214. The source endpoint 200 may conduct a call Quality of Service (QoS) reservation, such as Resource Reservation Setup Protocol (RSVP), for the media path 212. After the media path 212 is reserved, the source endpoint 200 sends out the media trace packets 203. The media trace packets 203 have essentially the same RTP header information
15 as conventional RTP media payload packets but with small TTL values that initially cause rejection by intermediate media path nodes.

 Because the headers 204 have similar information as conventional RTP packets, the media trace packets 203 travel over the same media path 212 as RTP packets carrying actual media payloads. The RTP header 204 also allows the media trace packets 203 to pass
20 through a firewall 216 to a subnet 219 and conduct further analysis for the remainder of the media path 212 between the firewall 216 and the destination endpoint 214.

 The endpoints 200 and 214 include any combination of hardware and software for receiving or sending a media stream over a media path 212 in the IP network 202. The endpoints 200 and 214 may comprise voice gateways that receive calls on behalf of other
25 devices. For example, the gateway can establish IP media sessions on behalf of Plain Old Telephone Service (POTS) phones, Private Branch Exchanges (PBXs), etc. Other endpoints can include Voice Over Internet Protocol (VoIP) phones or computers coupled directly to the IP network 202.

5 The media trace packets 203 look exactly like normal voice RTP packets with the same Differentiated Services Code Point (DSCP) bits, same Class of Service (CoS) bits, and same UDP ports. Except the media trace packets 203 have a different SSRC value because the time reference will be different. That is, the media trace packets 203 typically will not be sent at 10 millisecond (ms) intervals like conventional media packets.

10 When the media trace packets 203 have a sufficiently-large TTL value, the media trace packets will be received by the destination endpoint 214. When the destination endpoint 214 receives the packet, it must respond to the source endpoint 200. With traditional UDP traceroute, this response is typically an ICMP 'port unreachable' error, which indicates to the source endpoint that the TTL value is large enough to communicate with the
15 remote station. However, because the destination endpoint 214 is actually listening for incoming packets, it won't generate the traditional ICMP 'port unreachable' error, and must provide a different indication. This different indication is described in U.S. Patent Application Serial Number 10/723,118, filed November 26, 2003 which is incorporated by reference.

20 In one example, the media trace packets are sent with small incrementally increasing TTL values in the TTL field. This causes each intermediate router in the media path 212 to send back an Internet Control Message Protocol (ICMP) "time exceeded" error code when the router decrements the TTL value to zero. Multiple packets with the same TTL value can also be sent to measure jitter up to the point where the media trace packet is rejected in the
25 network.

 For example, referring to FIG. 2, the source endpoint 200 sends out a first media trace packet 220 having a TTL value of "X". The TTL value X is typically a small value that can only be processed by one or a few network processing devices before the media trace packet

5 220 is rejected and an error notice sent back to the source endpoint 200. For example, the source endpoint 200 may start with a TTL value of $X=1$. Every several media trace packets, the source endpoint 200 may increment the TTL value by one ($TTL=X+1$), until a particular TTL value is sufficiently large to reach the destination endpoint 214. For the remainder of the explanation below, it is assumed that the initial TTL value is $TTL=1$, and the TTL value
10 is incremented by one every predetermined number of media trace packets.

The first media trace packet 220 with $TTL=1$ is sent over the media path by the source endpoint 200. The first network node 226 receiving the media trace packet 220 decrements the TTL value $TTL=1$ by one. Because the TTL value is now zero, the network node 226 discards the media trace packet 220 and sends a reject notice 225 back to the source endpoint
15 200. The reject notice 225 may contain a portion of the media trace packet 220 payload. This allows the source endpoint 200 to correlate the reject notice 225 with specific sent media trace packets.

The source endpoint 200 may send out more than one media trace packet 220 with the same TTL value $TTL=1$. The source endpoint 200 can then compare the response time for
20 each of the multiple media trace packets 220 with the same TTL value to analyze the jitter between the source endpoint 200 and the first network node 226.

The source endpoint 200 then sends one or more media trace packets 222 at a next incremented TTL value. In this example, the TTL value is incremented by one to the value $TTL=X+1 = 2$. The source endpoint 200 sends out a first one of the media trace packets 222
25 with value $TTL=2$.

The first network node 226 decrements the media trace packet 222 by one to $TTL=1$. The network node 226 could decrement the TTL value of media trace packet 222 by more than one if there is a large amount of processing delay. Therefore, it is possible, that the

5 network node 226 could decrement the TTL value $TTL=2$ to zero. In this case, the network node 226 would also send a reject notice back to the source endpoint 200 for the media trace packet with an initial TTL value $TTL = 2$. If the source endpoint 200 receives reject notices back from network node 226 for $TTL = 2$, a network problem is identified for network node 226.

10 However, in a properly operating system, the first network node 226 decrements the TTL value in media trace packets 222 by one to the value $TTL=X-1=1$, and forwards the media trace packet 222 to the next hop in the media path. In this example, a second network node 232 receives the media trace packet 222 and decrements the TTL value a second time to $TTL=X-2=0$. Because the TTL value is now zero, the network node 232 sends back a reject
15 notice 230 back to the source endpoint 200 for the media trace packet 222.

The source endpoint 200 then sends out a next set of media trace packets 224 with a next incremented TTL value $TTL=3$. This process is repeated until a set of media trace packets reaches the destination endpoint 214. Thus, QoS information is obtained for each link and node in the media path between the source endpoint 200 and the destination endpoint
20 214.

FIG. 3A shows one example of how the media trace packets are used to analyze and modify a media path. In block 240, the source endpoint, such as a VoIP phone, gateway, server, router, switch, etc., receives a triggering event. The triggering event may simply be a timer that causes the endpoint in block 242 to automatically start a media path analysis
25 session. For example, the endpoint may be programmed to conduct the analysis session late at night when there is less traffic on the IP network. Alternatively, the triggering event in block 240 may be a network administrator manually typing a command that initiates the media session in block 242.

5 In block 244, a media path is established with a destination endpoint. This may include conducting a RSVP reservation or conducting some other type of QoS reservation protocol. After the media path is established, the TTL values in the RTP payload packets are modified in a variety of different possible ways in block 246 for analyzing the media path. For example, the TTL technique shown in FIG. 2 could be used for media path analysis.

10 However, other ways of varying the TTL values can also be used. The media trace packets are sent out on the IP network in block 248.

 The responses from the intermediate nodes in the media path are analyzed in block 250. A predetermined sequence of TTL values could be used in block 246 in the media trace packets and then the responses recorded for analysis by the network administrator.

15 Alternatively, in block 246 a program could dynamically modify the TTL values according to the responses from previously sent media trace packets analyzed in block 250. In another embodiment, the network administrator could manually modify the TTL values in block 246 according to the responses received in block 250.

 For example, if a particular intermediate network node sends back reject notices for

20 an unexpected TTL value, extra media trace packets could be sent at the same TTL value to verify the network response. A set of media trace packets could also be sent at a very large TTL value to determine if the same intermediate network node continues to send back rejection notices.

 According to the automatic or manual analysis conducted in block 250, the preset

25 media path taken by the media streams may be either manually or automatically reconfigured in block 254. For example, all further media calls may be initiated through a different Internet Service Provide (ISP) when the rejection notices from the current media path

5 indicate excessive delay, jitter, dropped packets, indirect media paths to a destination endpoint, etc.

FIG. 3B shows another application where the media trace packets are sent out at the same time that actual media payload packets are being exchanged between two endpoints. A conventional media session for transporting an actual media stream is initiated in block 260.

10 For example, a user may dial a destination phone number that initiates the media session. A media path for the media session is established in block 262. A conventional media session is then conducted between two endpoints in block 264. For example, the two endpoints may conduct a VoIP phone call, or a media server may provide an audio and/or video media stream to a requesting endpoint, etc.

15 In block 266 one or both of the endpoints monitor the actual media packets for a triggering event. A triggering event for example may be a particular number of dropped media payload packets, a large packet delay, or a large amount of media packet jitter detected between the two endpoints. In one example, contents of Real Time Control Protocol (RTCP) reports are used to identify triggering events. If no triggering event is detected in block 266,
20 conventional media payload packets continue to be transferred in block 264. Other ways of identifying triggering events are known to those skilled in the art and therefore are not described in further detail.

If one of the triggering events is detected in block 266, media trace packets are generated in block 268. The TTL values and SSRC values for the media trace packets are
25 modified from the conventional media payload packets. The media trace packets are then sent out either in conjunction with the actual media payload packets in block 270.

Alternatively, the media trace packets are sent out after the media call has finished or when

5 there is a detected silence period or non-transmission period in the current media session.
The responses to the media trace packets are analyzed in block 272.

The analysis for the media trace packet responses may be recorded for future review by the network administrator. The media session with the actual media payload packets is then continued in block 264. Alternatively, if the analysis in block 272 indicates a media
10 path problem, a new media path may be established either for the present call or for future calls in block 274. For example, new media sessions may be routed around an identified network node. The media sessions are then continued in block 264.

FIG. 4 shows the TTL field 206 in an IP packet 203. The IP packet 203 consists of a IP header 239, UDP header 241, and an RTP header 243. RTP is carried in UDP which is
15 carried in IP. The IP fields in the IP header 203 are described in RFC791, the fields in the UDP header 241 are described in RFC 768 and the fields in the RTP header 243 are described in RFC 3550 which are all herein incorporated by reference.

The TTL field 206 is manipulated as described above for conducting the media path analysis. The value in the TTL field 206 indicates the maximum time the datagram is
20 allowed to remain in the internet system. When this field is decremented to the value zero, then the datagram is destroyed. The TTL field 206 is modified during internet header processing.

The time is measured in units of seconds, but since every module that processes a datagram must decrease the TTL by at least one even if it processes the datagram in less than
25 a second, the TTL value should be thought of only as an upper bound on the time a datagram may exist.

A network node processing the media trace packet does not send a datagram with a TTL value of zero and does not discard a datagram just because it was received with a TTL

5 value less than 2. The IP layer provides a means for the transport layer to set the TTL field
206 of every datagram that is sent. Expiration of the TTL value will cause a datagram to be
discarded by a gateway but not by the destination host. However, hosts that act as gateways
by forwarding datagrams may follow the gateway rules for TTL.

Thus, the media trace packet 203 allows the sender to determine the currently-used
10 path for the RTP flow and the delay and jitter contribution along the path. Alternatively, the
endpoint (IP phone, Public Switched Telephone Network (PSTN) gateway, dedicated
diagnostic device, etc.) could send the media trace packet 203 at the beginning or end of a
call to collect information about the underlying network.

The media trace packet 203 works through firewalls and Network Address Translators
15 (NATs), since there are already pinholes for the UDP ports for the RTP stream. The media
trace packet 203 is also guaranteed to use the same path as the RTP stream.

The media trace packet 203 can be used in VoIP phones with user initiation or with
automatic initiation and in IP gateways with automatic initiation via signaling or Digital
Signal Processing (DSP) error thresholds. This analysis technique is also useful for a
20 dedicated diagnostic device, or a device possibly integrated into a router which generates
synthetic calls. Low volume noise may be inserted into the media trace packet payloads so
that any media trace packets reaching the destination endpoint do not annoy the listener.

Any exposure to Denial of Service (DoS) attacks for the RTP packets would be the
same as current RTP packet attackers. Mitigation of the DoS attacks could be provided by
25 rate limiting ICMP, or rate limiting packets with abnormally-small TTL values.

Using Media No-Op Packets for Media Path Tracing

5 The media trace packets can be used in combination with a media payload type referred to as a no-op payload packets or simply no-op packet. One application for the no-op packet is in the Real Time Transport Protocol (RTP), which is specified in RFC3550. The no-op payload packet, when used in RTP, is referred to as an RTP no-op packet. The no-op packet in one embodiment has no media payload and uses a one-bit opcode in the RTP
10 payload for requesting a media path continuity report. The TTL fields in the no-op packet can be varied as described above in FIGS. 1-4 in order to analyze intermediate nodes and links in the media path.

 The no-op payload packets are literally a no-op coder. When negotiated through a Session Description Protocol (SDP – RFC2327), H.245, or some other media description
15 protocol, any sender is able to send RTP packets with this coder. Any receiver expressing a willingness to receive this coder processes the RTP no-op packets according to the basic RTP specification and then discards the RTP no-op packets without performing actual media play-out.

 The RTP no-op packets allow the instrumentation mechanisms in RTP, namely the
20 information in Real Time Control Protocol (RTCP) reports, to be evaluated prior to actually sending voice, audio, or other media in an RTP session. For example, an implementation may decide that the media stream is not sent for this session if the RTCP report indicates a low quality media path.

 FIG. 5 shows a Wide Area Network (WAN) or Local Area Network (LAN) referred
25 to generally as IP network 326 that is used for transferring a real time media stream 338. Plain Old Telephone Service (POTS) phone 312, Private Branch Exchange (PBX) 311A, etc. are coupled to the IP network 326 through gateway 316. Another POTS phone 332, PBX 311B, etc. are coupled to the IP network 326 through gateway 330. Voice Over Internet

5 Protocol (VoIP) phones 310A and 310B and computers 314 and 334 are coupled directly to the IP network 326.

Any of the voice gateways 316, 332, VoIP phones 310A, 310B, and computers 314 and 334 include any combination of hardware and software for receiving or sending a media stream 338. The media stream 338 can include any real time media such as audio data, video
10 data, and real time text such as the text used for the hearing impaired.

Any combination of the phones 310A, 310B, 312, and 332; PBXs 311A and 311B; computers 314 and 334; and gateways 316 and 330 are referred to below generally as endpoints, users, senders or receivers, etc. One of the endpoints initiates a media call to another endpoint. If one of the phones 312 or 332 initiates the call, the gateway 316 or 330,
15 respectively, would make the VoIP call on behalf of the phone. For example, the phone 312 may make a VoIP phone call to phone 332. A caller dials a phone number using phone 312 and the gateway 316 then determines the IP address for the gateway 330 serving phone 332. The two gateways 316 and 330 then conduct a signaling protocol for establishing the VoIP call between phone 312 and phone 332.

20 In another scenario, the computer 314 may want to receive a video stream or audio stream from computer 334 which operates as a music and video content server. In another scenario, the computer 314 may wish to make a VoIP call to VoIP phone 310B. The computer 314 initiates the media connection to VoIP phone 310B. The computer 314 may establish the media connection through a gateway or may access VoIP phone 310B through
25 the IP network 326, without using a gateway. Media connections can also be initiated from any of the other devices 310A, 310B, 311A, 311B, 332, or 334.

In the example described below, the phone 312 initiates a VoIP phone call to phone 332. For clarity, operations will be described as being performed by the phones 312 and 332.

5 However, some of these operations could be performed by the gateways 316 and 330. The phone number for phone 332 is entered at phone 312. There may be an intermediary device, such as a call controller that determines the IP address for gateway 330 or phone 332 based on the phone number dialed by phone 312. During this initial signaling, the two gateways 316 and 330 agree to use the no-op payload packet verification process described below.

10 Prior to transmitting the media stream 338, gateway 316 sends one or more no-op payload packets 322. In this example, the no-op payload packets are RTP no-op packets and three of the RTP no-op packets are sent from gateway 316 to gateway 330. However any type of no-op payload packets 322 can be sent depending on the media stream carrier protocol. Any number of the no-op payload packets can be sent by gateway 316. However,
15 the number of no-op packets should be limited to prevent substantial delay in completion of the call setup.

The RTP no-op packets 322 have the same format as conventional RTP packets except that the packet payload contains no media. In the example shown, a marker bit 324 in the packet payload is not set in the first two RTP no-op packets 322 but the marker bit 324 is
20 set in the third RTP no-op packet 322. It should be understood that this is only one example and any number of no-op packets 322 can be sent. For example, in other implementations less than three, or more than three, no-op packets are sent with the last no-op packet having a set marker bit 324. In another example, only one no-op packet is sent with that single no-op packet having a set marker bit.

25 When the gateway 330 receives the RTP No-Op packet with the marker bit set, the gateway 330 immediately generates an RTCP report 328 that includes Quality of Service (QoS) data for all of the RTP no-op packets 322 previously received. The RTCP report 328 is generated in the same manner used for generating RTCP reports for conventional RTP

5 packets containing media payloads. For example, the phone 332 may count the number of received RTP no-op packets 322 as well as tracking latency and jitter statistics.

The RTCP report 328 would normally be sent by gateway 330 to gateway 316 about every 5 seconds. However, the set marker bit 324 in the RTP no-op packet 322 causes the gateway 330 to immediately send the RTCP report 328 to gateway 316. In this example, the
10 RTCP report 328 would indicate that three RTP packets were received by gateway 330 if no packets were lost or unduly delayed. If the information in RTCP report 328 indicates the media path 318 carrying the RTP no-op packets 322 is operating adequately, signaling 336 is sent causing phone 332 to ring. This can be the un-modified "alert" signaling, or could be QoS-aware signaling using preconditions as described in RFC3312. Then the two phones 312
15 and 332 start sending the real time media stream 338.

If the RTCP report 328 indicates that the media path 318 is not operating adequately the signaling protocol is so informed, and can take action based on local or global policy. Possible policies include, but are not limited to: aborting the call such that the phone 332 does not ring, continuing with the call but making note to the accounting and billing system
20 that the call may have to be discounted due to poor quality, or eliminating the problematical media stream if the call carries multiple media streams (e.g. audio and video).

In the case where the call is aborted, the aborted call may be established over an alternate path in IP network 326 or over an alternative medium, such as over a circuit switched network. A poorly performing media path may be identified when the RTCP report
25 328 indicates that fewer RTP no-op packets were received by gateway 330 than were sent by gateway 316. A media path problem may also be identified in the RTCP report 328 if the amount of latency or jitter in the RTP no-op packets 322 is outside of some expected range. In another scenario, a bad media path is indicated when the RTCP report 328 is not received

5 by gateway 316 within some amount of time after gateway 316 sends out an RTP no-op packet 322 with the set marker bit 324.

The no-op packet call verification process described above can be performed independently for each endpoint. For example, in a bidirectional transmission, such as with a phone call between phones 312 and 332, each phone can make a determination whether or
10 not to establish the call based on the information in the RTCP report 328 received from the other phone or gateway, or the lack of a RTCP report 328.

If the media transmission is unidirectional, then only one endpoint may need to verify the media channel. Unidirectional media transmissions are used for music or video streams transmitted unidirectionally from a media server to a user. In this application, only the path
15 from the media server to the user is important for media quality considerations. Thus, only the user receiving the media stream needs to send RTP no-op packets and evaluate the RTCP report 328 sent by the media server.

Modifying Time To Live In RTP No-OP Packets

20 The RTP no-op packets 322 can serve the dual purpose of both indicating an acceptable media path between two endpoints as described above in FIG. 5 and can also be used for analyzing intermediate links and nodes in the media path 318 as described above in FIGS. 1-4.

The TTL field 206 in no-op packets 322 can be changed to low values that would
25 cause the no-op packets 322 to be rejected by intermediate nodes in media path 318. The endpoints ending the no-op packets 322 could then analyze the rejection notices by the intermediate nodes to isolate any problems in the media path 318.

5 In one example, the source endpoint sends out the RTP no-op packets 322 with a relatively large value in TTL field 206. The TTL value is set large enough so that in a normal operating condition the no-op packets 322 reach the destination endpoint before being decremented to a zero value by an intermediate network node. For example, the TTL value may be set to a standard preconfigured value of 32 or 64. In a normal operating condition,
10 the destination endpoint receives the RTP no-op packets 322 and sends back an associated RTCP report 328. If the RTCP report 328 indicates a properly operating data path 318, the source endpoint starts sending the media stream 338.

However, if the RTCP report 328 indicates a problem with the media path 18, the source endpoint can start modifying the TTL values 206 in no-op packets 322 in order to
15 isolate the media path problem. For example, the RTCP report 328 may indicate that some of the sent no-op packets 322 were not received or that there was an unusually large delay in receiving the no-op packets.

The source endpoint then isolates the cause of the media path problem by varying the TTL values in following no-op packets. For example, the TTL value in the next sent no-op
20 packets may be set to one. The source endpoint then waits for an expected rejection notification from the first network node in the media path. The source endpoint can then continue to increment the TTL values by one until the source of the media path disruption is identified. The source endpoint could then either manually or automatically reroute calls around the identified network problem.

25 FIG. 6 shows generally how the initial call signaling is conducted between two endpoints. The Request For Comment (RFC) 3312 specifies one example of a real time media signaling protocol. Of course, other types of real time media stream signaling protocols can also be used. Examples of signaling protocols include Session Initiation

5 Protocol (SIP), Session Definition Protocol (SDP) offer-answer, H323, and Media Gateway Control Protocol (MGCP).

 Signaling messages 342 are sent from phone 312 to phone 332 prior to transmitting the media stream 338 shown in FIG. 5. The signaling messages 342 include a request or notification that the phone 312 intends to conduct a media path continuity check using RTP
10 no-op packets. The signaling 342 may need to be conducted through a call controller 340. The phone 332 agrees to conduct a continuity check using the RTP no-op packets by sending acknowledgement signaling 344 back to phone 312.

 The phone 332 can also send additional notification back to phone 312 in signaling 344 that indicates phone 332 also intends to send RTP no-op packets to phone 312 for
15 continuity checking. The two phones 312 and 332 thus both agree to send and receive the RTP no-op packets 322 (FIG. 5) prior to sending and receiving a media stream.

RTP No-Op Packet Sender

 FIG. 7 shows in further detail the sequence followed by the RTP no-op packet sender.
20 In block 350, sequence numbers are assigned to the RTP no-op packets starting at the next sequence number for the RTP session. The sequence number for each no-op packet is incremented by one in accordance with the RTP specification. In block 352, a timestamp is inserted into the no-op packet according to a packet time (ptime) parameter, also in accordance with RTP operation. The ptime parameter is conveyed via the signaling protocol,
25 for example via SDP. It denotes a packetization interval identifying how many milliseconds of media are encoded into a single RTP packet.

 The sender identifies in block 354 the last RTP no-op packet to be sent. If it is not the last no-op payload packet to be sent, then the sender transmits the no-op packet in block 358.

5 If it is the last no-op packet, then the sender sets the marker bit in the RTP header of the no-op payload packet in block 356 causing the no-op packet receiver to immediately send back an RTCP report. The RTP no-op packet with the set marker bit is then transmitted to the receiver in block 358.

The RTCP report is sent within the session RTCP bandwidth limits and other RTCP
10 sending rules. The scheme for sending the RTP no-op packets from the opposite end of the call are the same.

RTP No-Op Packet Receiver

FIG. 8 shows in further detail the operations used at the processing device receiving
15 the RTP no-op packets. In block 360, a syntax check is performed on the received RTP no-op packets in the same way as any other RTP payload. In block 362, jitter buffer operations are performed on the RTP no-op packets in the same manner as any other RTP payload. The jitter buffer operations may include Synchronization Source field (SSRC), sequence, and timestamp processing.

20 In block 364, the RTCP statistics are updated for the RTP no-op packets in the same manner as for any other RTP payload format. If the marker bit in the RTP no-op packet is not set in block 366 or other RTCP sending rules are not met in block 367, the receiver continues to update RTCP statistics for any additionally received RTP no-op packets. If the marker bit is set for a received RTP no-op packet in block 366, other RTCP sending rules are
25 checked in block 367. For example, it is determined if the RTCP bandwidth constraints allow sending the RTCP report. If these RTCP sending rules are met in block 367, the RTCP report is immediately sent to the no-op packet sender in block 68. The receiver rules for the other endpoint are the same.

Media Channel Acceptance

FIG. 9 describes the offer-answer exchange between the two endpoints. In block 370, one of the endpoints proposes using the "no-op" coder and payload format during the session.

10 In one example, an SDP (or H.245) with an offer-answer style exchange is used for conducting the RTP no-op notification.

In block 372, the notification may also optionally carry an RFC3312 precondition to ensure that no user is alerted of a call before the acceptance procedure using the no-op payload packets is successfully completed. The preconditions could be used in either
15 direction, although it may be typical to enforce their use in both directions. For example, when there is an RTP session in both directions of a full-duplex VOIP call, both directions would be required to pass the no-op packet acceptance test.

After completing the first RFC3312 offer-answer exchange, the sender starts sending a configurable number of RTP no-op payload packets to the receiver in block 374. The
20 marker bit is set in the last no-op payload packet in block 376. The receiver conducts the RTCP processing and returns the RTCP report after seeing the set marker bit. If the packet with the marker bit was lost, the RTCP report may be sent after the normal startup RTCP delay interval.

The sender waits for the RTCP report in block 378. If the RTCP report is not
25 received in block 386 within some time period after sending out the last RTP no-op packet (i.e., timeout), the media path is declared unacceptable and the signaling protocol can decide what to do in block 388 according to any range of policies such as the ones described above.

5 Alternatively, the procedure described above is repeated to cover the case of a single RTCP report packet getting lost.

If the RTCP report is received in block 378 within an acceptable time period, the RTCP report is examined in block 382. Depending on the loss and jitter statistics, the receiver either declares the channel acceptable or unacceptable in block 382. If the media
10 channel is deemed unacceptable, the signaling protocol can decide what to do in block 388 according to the predetermined range of policies. If the channel is deemed acceptable, the precondition is marked as satisfied and the sender initiates, or responds to, the second phase of call establishment in block 384 that allows the call to proceed. In particular, one of the endpoints may notify a user of the call by activating an annunciation device.

15 Referring to FIG. 10, the RTP no-op packet 322 is formatted to look like a normal RTP payload packet. This allows the no-op packet to operate and be processed in the same way as a voice, video or other type of RTP media packets. The RTP no-op packet will also take the same path as the RTP packets that are sent after the call is set up. The only difference is that the RTP no-op packets 322, 323, 324 are not played out by the receiver.

20 The packet header 389 includes conventional RTP header content including a sequence number, timestamp value, synchronization source identifiers and contributing source (CSRC) identifiers. The timestamp value can be the real time-stamp used with actual media payloads or the timestamp value can be simulated.

The RTP no-op payload 387 employs the marker bit 324 in the RTP header. If bit 0 is
25 set, the receiver immediately sends out an RTCP report. Bits 1-31 in the payload 387 are reserved and in one example must all be set to zero. Additional padding bytes may be appended up to the negotiated packet time (ptime) value in SDP. Padding may be useful to generate RTP packets that are the same size as other payload, such as a normal voice payload.

5 Of course other formats can also be used in the RTP packet 322 for generating the no-op payload format. If the no-op packet 322 is also used as a media trace packet, the TTL field 206 (FIG. 4) in the IP header for the RTP packet 322 is modified as described above.

FIG. 11 shows in further detail the elements in a network device 390 that sends or receives the RTP no-op packets and the media trace packets. The device 390 can provide the
10 functionality in any combination of phones 312 and 332, computers 314 and 334, and gateways 316 or 330 shown in FIG. 5. For example, in one configuration the device 390 is either computer 314 or 334. The computer 314 or 334 performs all of the operations used for conducting the media session, including dialing another telephone or accessing another computer. In another configuration, the device 390 may be the gateway 316 or 330 shown in
15 FIG. 5 and the media session is initiated by a VoIP phone or computer that is connected to the gateway.

A processor 392 is coupled to a user interface 394 and a network interface 396. The network interface 396 connects to the IP network 326. The user interface 394 connects to any device needed to initiate the media connection. For example, user interface 394 could
20 connect to a keyboard or to a phone.

A memory 398 contains executable software that contains both call signaling software 400 and RTP and RTCP software 402. The memory 398 represents any combination of internal memory within the processor 392 or memory external to the processor 392. For example, the memory 398 can be any combination of flash, Read Only Memory (ROM),
25 Random Access Memory (RAM), or disk memory used for containing and executing the signaling software 400 and the RTP and RTCP software 402.

The signaling software 400 is used for establishing the initial media path between the two endpoints. The signaling software 400 includes the RTP no-op negotiation described

5 above. The RTP and RTCP software 402 conducts conventional RTP and RTCP with the addition of conducting media path verification using the RTP no-op packets and responding with RTCP reports as described above.

Thus, the RTP no-op payload format, with the special use of the RTP marker bit and RFC3312-style offer-answer and call signaling preconditions are used to perform channel
10 acceptance testing as part of VoIP call establishment. Standard RTP packets are used rather than something special like a ping or an NSE. The marker bit is used along with RTCP to control the feedback loop.

This no-op payload scheme solves the problem of assessing media connectivity and channel quality in packet switched networks. The no-op scheme can be used in any and all
15 VOIP or media endpoints, but particularly those for sale to and through service providers accustomed to channel assessment capabilities, such as those provided in circuit switched networks. The no-op payload format can also be used for other applications, such as synthetic load testing and to foil traffic analysis.

The system described above can use dedicated processor systems, micro controllers,
20 programmable logic devices, or microprocessors that perform some or all of the operations. Some of the operations described above may be implemented in software and other operations may be implemented in hardware.

For the sake of convenience, the operations are described as various interconnected functional blocks or distinct software modules. This is not necessary, however, and there
25 may be cases where these functional blocks or modules are equivalently aggregated into a single logic device, program or operation with unclear boundaries. In any event, the functional blocks and software modules or features of the flexible interface can be

5 implemented by themselves, or in combination with other operations in either hardware or software.

Having described and illustrated the principles of the invention in a preferred embodiment thereof, it should be apparent that the invention may be modified in arrangement and detail without departing from such principles. I claim all modifications and variation
10 coming within the spirit and scope of the following claims.